CS359 project report

System Overview

Purpose

The E199 Platform is a web-based emergency management system designed to coordinate responses to incidents like fires and accidents. It facilitates communication between civilians, volunteers, and administrators while managing emergency incidents effectively.

Technology Stack

- Frontend: HTML5, CSS3, JavaScript, Bootstrap 5.3.3
- Backend: Java (Servlets, REST APIs)
- Libraries: jQuery 3.7.1, OpenLayers (for maps)
- Data Format: JSON for data exchange

User Roles

Admin

- System management
- Incident oversight
- Communication management
- Statistics monitoring

The figure below displays the Admin Dashboard. Admin can access the system by login and also they can log out when they finish their task. The roles of an Admin include Manage Incidents, Submit Incident, Chat, View Statistics, and View Incidents History.

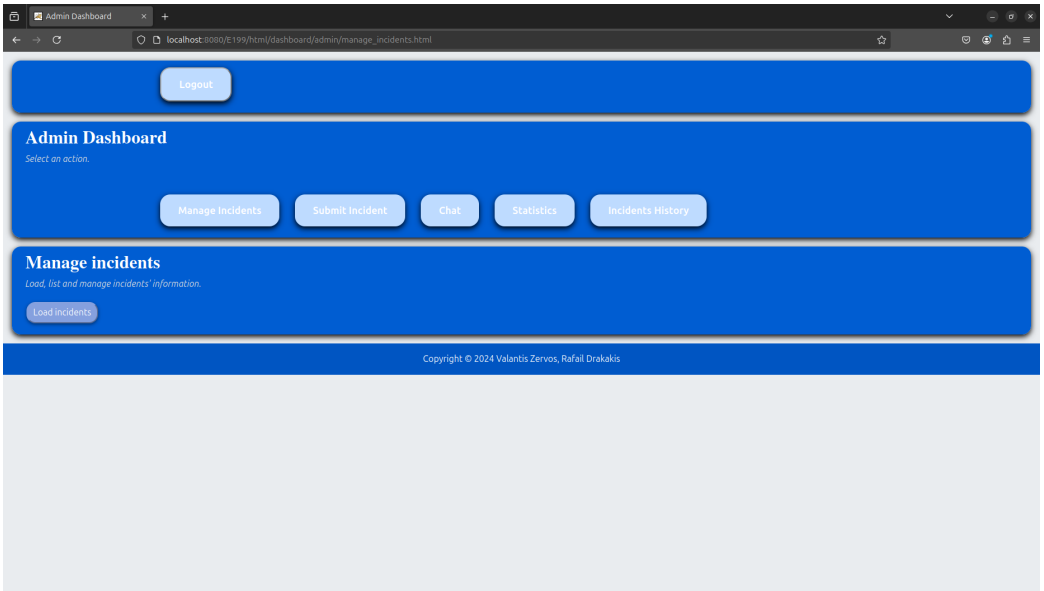

Figure 1: Admin Dashboard displaying some of the admin's roles

User

- Incident reporting
- Chat communication
- Profile management
- Incident history viewing

The figure below displays the User dashboard after they have access the system by login. The main activities included on the user side include viewing of notifications, submitting incidents, viewing of incidents and incidents history, chatting, and also viewing their personal information on their profiles.

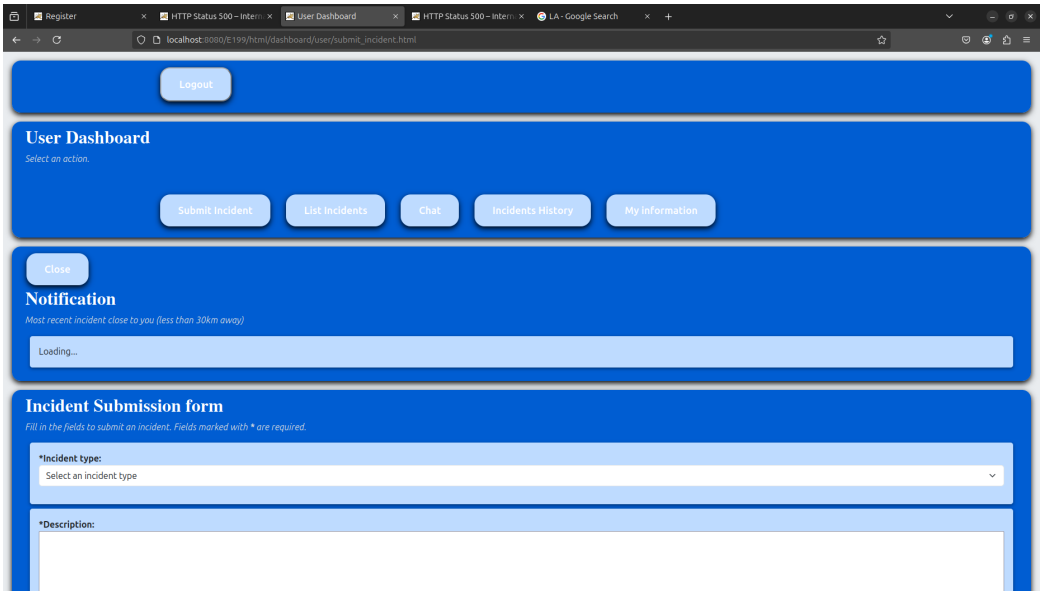

Figure 2: User Dashboard displaying some of the user's roles

Volunteer

- Incident participation
- Status updates
- Profile management
- Communication with admins

The figure below displays a Volunteer dashboard after they have access the system by login. Volunteers are users with limited functionalities as they cannot submit any incidents but from the dashboard they can view incidents and incidents history, chat, and also view their personal information on their profiles.

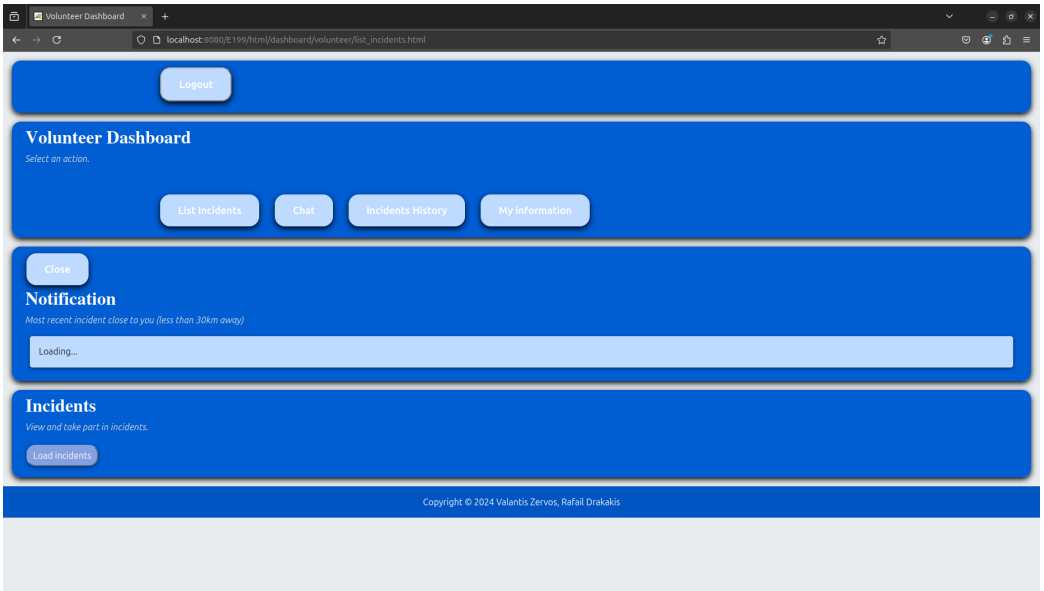

Figure 3: Volunteer Dashboard displaying some of the volunteer's roles

Guest

- Limited incident reporting
- Basic system access
- Public information viewing

The figure below displays a Guest user dashboard after they have access the system by login. Guest users are users without any account, they can access the system without registering or login. They are just one-time users of the system. Guest users are limited to submitting inclidents, viewing some incidents, and talking to the help desk.

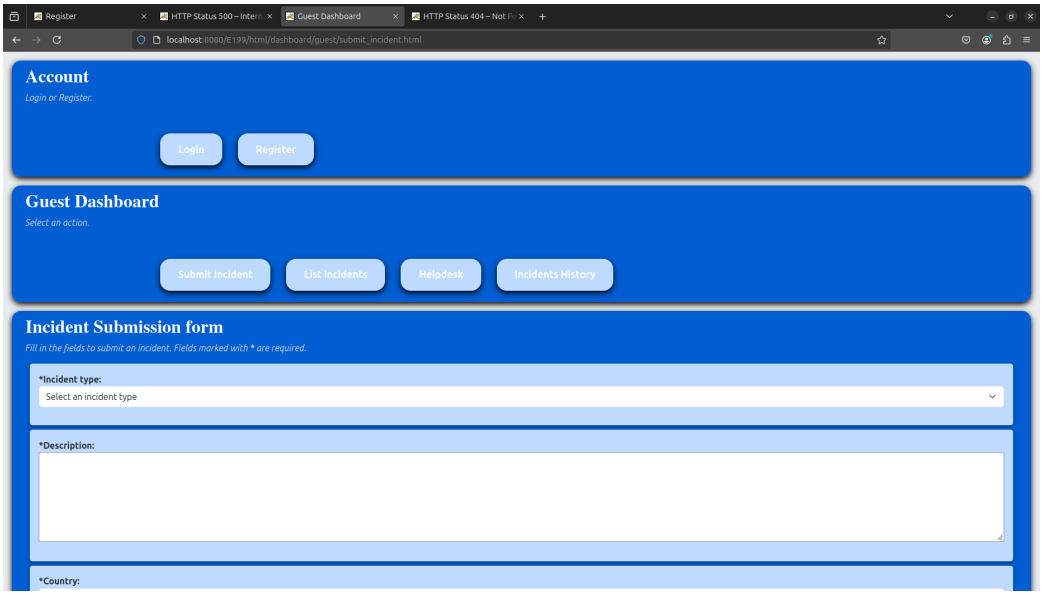

Figure 4: Guest Dashboard displaying some of the guest's roles

Implementation

Frontend Architecture

Each user type has a specialized dashboard with role-specific features:

```
/dashboard/admin/
chat.html                    # Admin communication interface
history.html                 # Incident history management
manage_incidents             # Incident oversight
statistics.html              # System analytics
submit_incident              # Incident creation
```

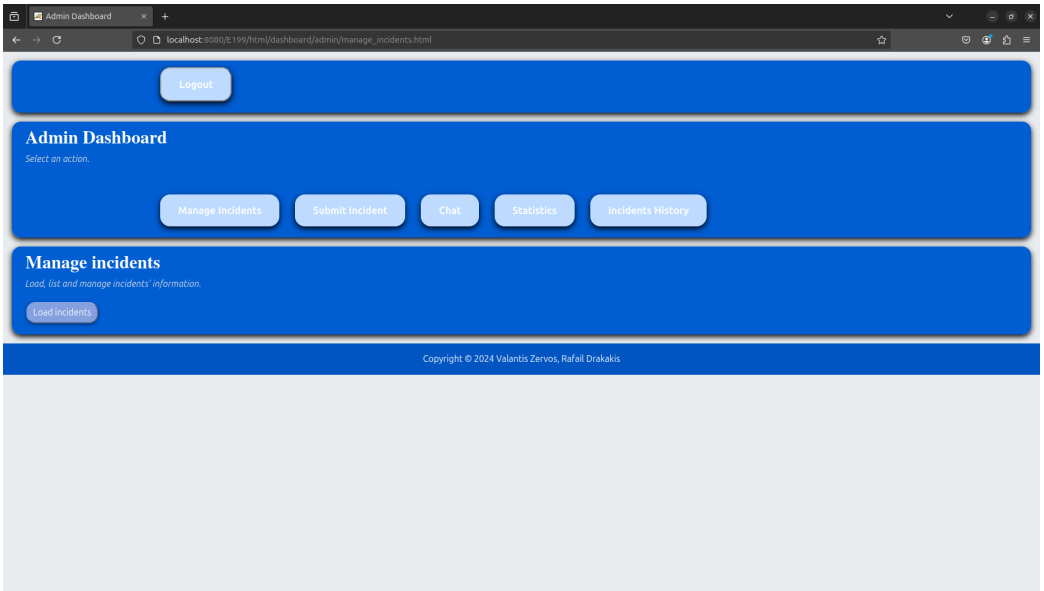

Figure 5: Admin Dashboard distinguishing available roles for the admin

The figure above displays the Admin Dashboard. An Admin is a super user and has access management of incidents reported by every user. The roles of an Admin include Manage Incidents, Submit Incident, Chat, View Statistics, and View Incidents History.

/dashboard/user/

```
chat.html          # User communication
history.html       # Personal incident history
list_incidents     # Active incidents view
my_information     # Profile management
submit_incident    # Incident reporting
```

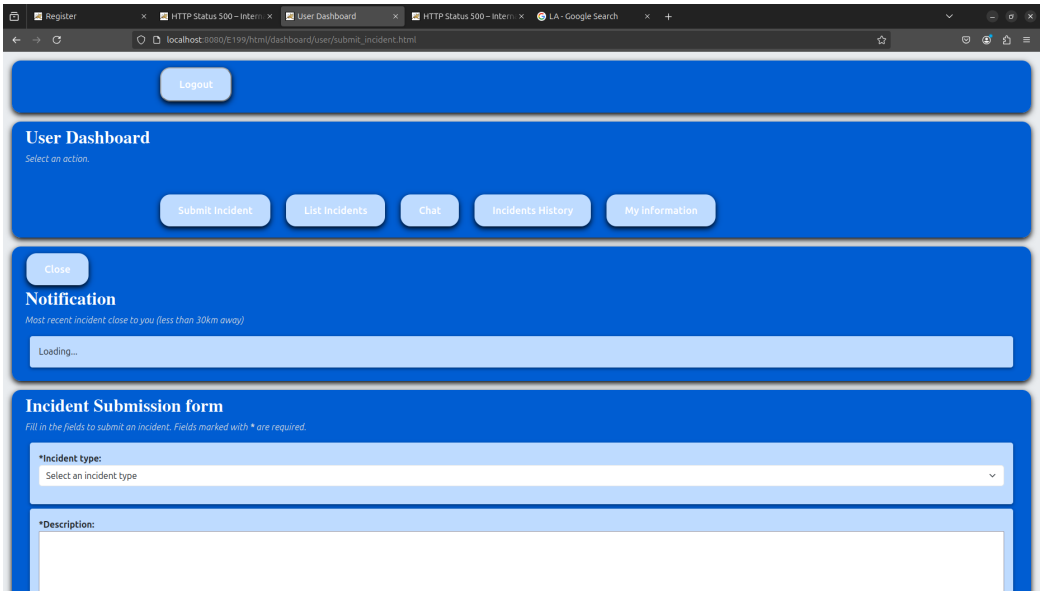

Figure 6: User Dashboard distinguishing available roles for the user

The figure above displays the User dashboard after they have access the system by login. The main activities included on the user side include viewing of notifications, submitting incidents, viewing of incidents and incidents history, chatting, and also viewing their personal information on their profiles.

Backend Architecture

Core Classes

```
mainClasses/

Admin.java          # Admin user management
Incident.java       # Incident data structure
Message.java        # Communication system
Participant.java    # Volunteer participation
User.java           # Base user functionality
Volunteer.java      # Volunteer management
```

REST API Structure

```
services/

API.java            # Base API functionality
RESTAPIDelete.java
RESTAPIGet.java
RESTAPIPost.java
RESTAPIPut.java
```

Key Features

Incident Management System

Incident Creation

Incident reporting is one of the major functionalities of the system. The creation of an incident can be done by almost any user, whether an admin, guest, or a typical user. The screenshot below displays a form used to create and submit an incident.

Features of Incident Reporting

- Type classification (fire, accident)
- Location mapping
- Resource requirement specification
- Priority/danger level assignment

Description of the Incident Submission Form

The user chooses the type of incident they would like to report (e.g., fire or accident). They can provide a brief description of the incident and also include more details, such as:

- Country
- Prefecture
- Municipality
- Address

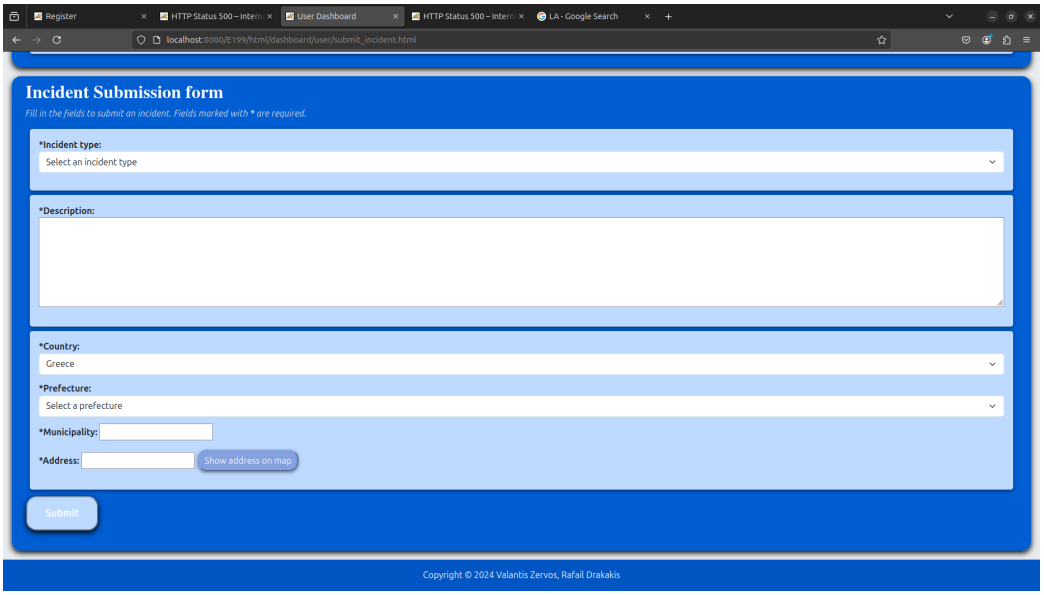

Figure 7: Incident Submission Form

Incident Tracking

- Real-time status updates
- Resource allocation
- Volunteer assignment
- Historical data maintenance

The reported incidents need to be tracked and the status of the incidents is recorded on the system. The volunteers can check active incidents and also update the status of the incident. Admins can as well manage incidents which are active and also the ones that have been attended to. They can also modify the status of an incident and view real time status of the incident.

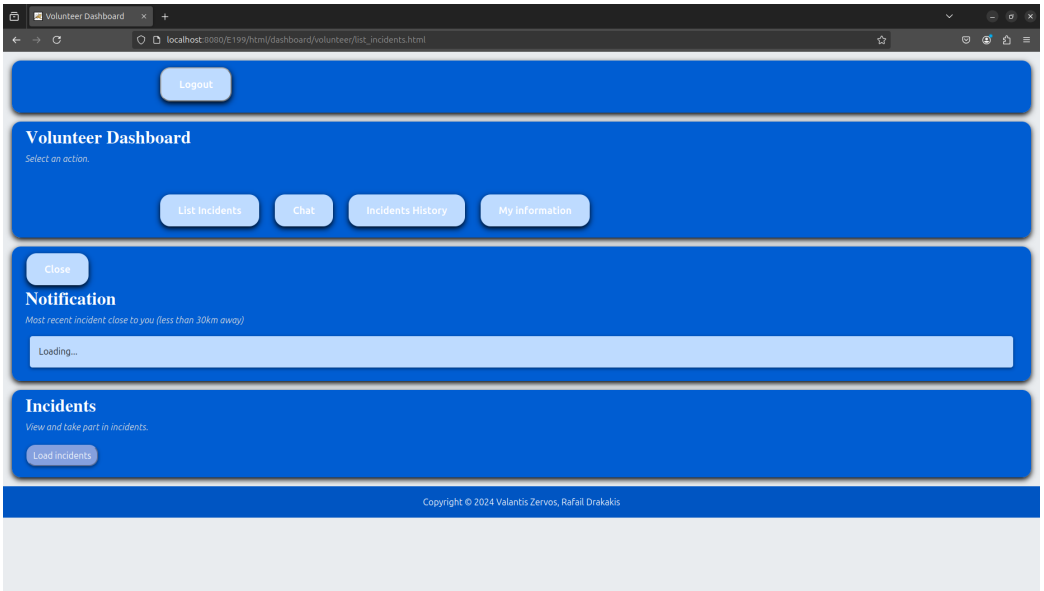

Figure 8: Volunteer Dashboard displaying incidents tracking section

Communication System

- Public channels
- Private messaging
- Volunteer coordination
- Admin broadcasts

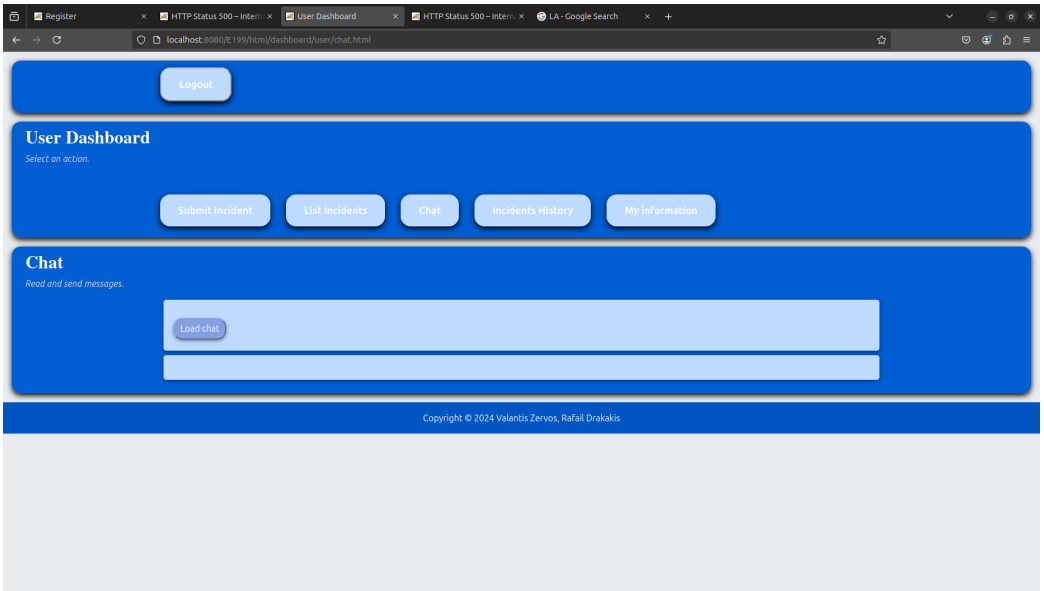

Figure 9: User Dashboard displaying chat section

Chatting is also a major functionality of the system. Users can send messages to one another and also to channels. The chat feature is available to admins, users and volunteers.

User Management

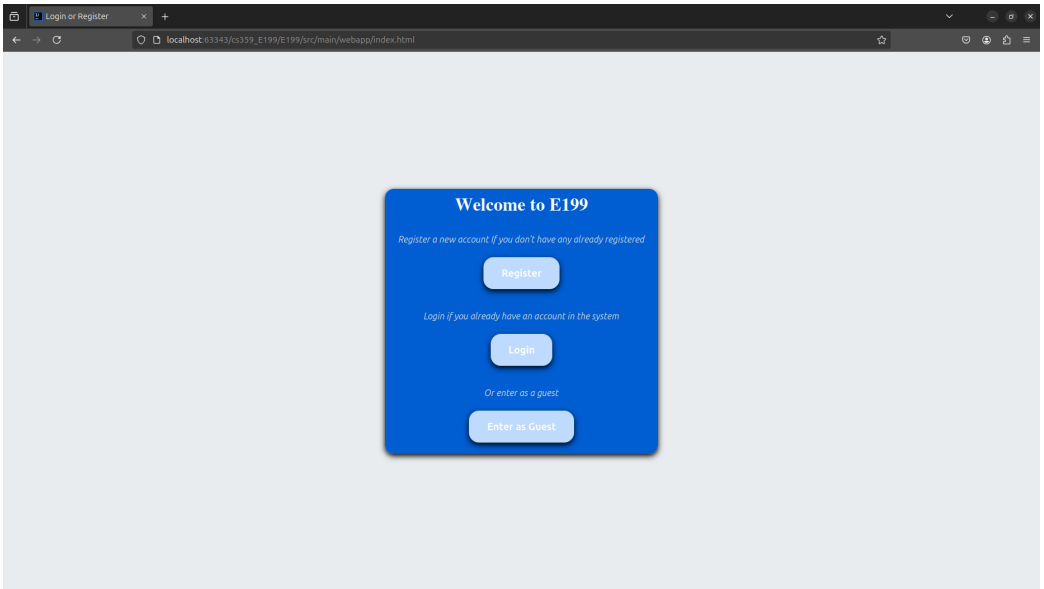

Figure 10: Welcome screen

The figure above displays the welcome screen of the application. New users can choose to register on the system to acquire an account. Existing users would be required to login to access the system. Guest users are not required to create an account, they can access the system and perform some limited functions.

Registration System

- Role-based registration
- Validation checks
- Profile customization
- Credential management

Figure 11: Registration form

New users of the application will need to click on the register button the welcome screen to access the registration screen. When registering, users need to provide valid personal information as the system has some validation checks. New users will choose their roles (simple user or volunteer) and this will classify them in role-based accounts. A simple user cannot login as an admin or a volunteer.

Authentication

```
session/
```

```
LoginUser.java
LogoutUser.java
GetActiveSession.java
```

Login Process

Registered users need to click on the login button to access the login screen. The login process involves the following steps:

- Users select the appropriate user type: Admin, Volunteer, or User.
- Users enter their username and password that they registered with.
- The system validates the entered credentials.
  - If the credentials are valid, the user will be logged in.
  - If the credentials are invalid, the system will display an error message on the screen.

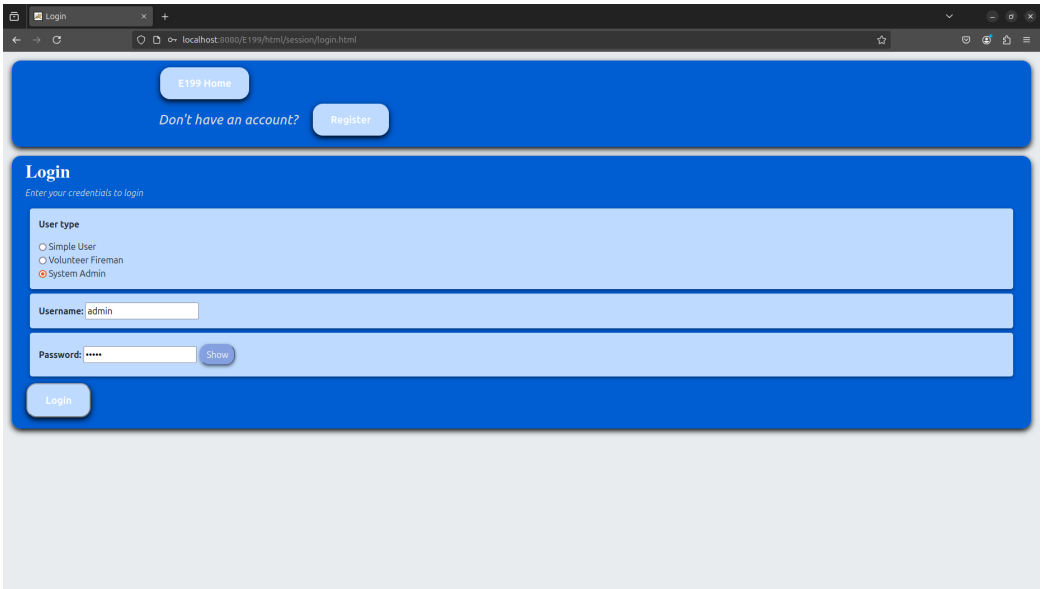

Figure 12: Login form

Volunteer Management

Volunteer Types

- Simple volunteers
- Driver volunteers
- Specialized roles

Participation System

- Request management
- Status tracking
- Performance evaluation
- Success metrics

Security Features

Authentication

- Session-based authentication
- Role-based access control
- Secure password handling

Data Validation

```
validation/
IsEmailAvailable.java
IsTelephoneAvailable.java
IsUsernameAvailable.java
```

Test Case Scenario

Testing the functionality will perform a simulation of authenticating one of the volunteers into the system and view some of the features on their dashboard. For the test case, we will simulate the process of registering a new user as a volunteer and afterwards login to the application with the credentials registered. The user

will access the welcome screen first then clicks on the register button and will be presented with a register page. Below is the screen displaying the new user registration process.

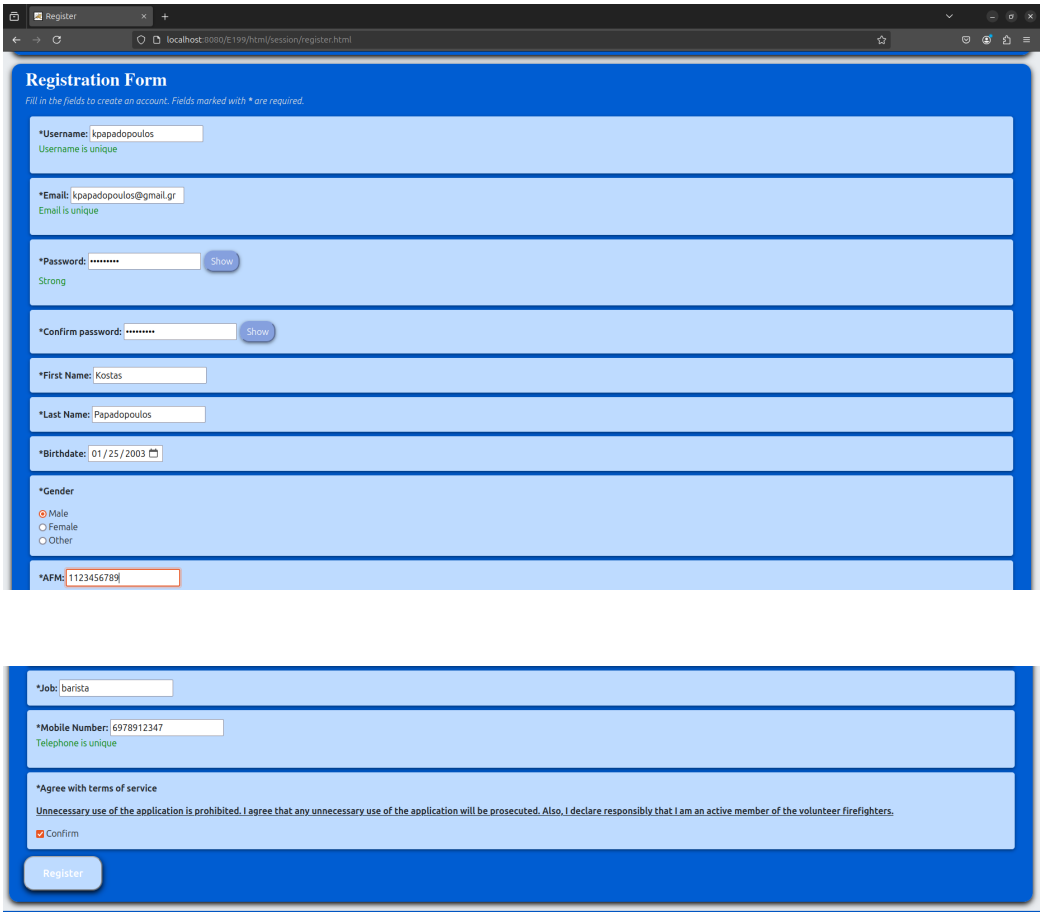

Figure 13: User registering his information on the application

After entering all valid information and clicking the Register button, a feedback screen is displayed to the user saying that the registration process was successful

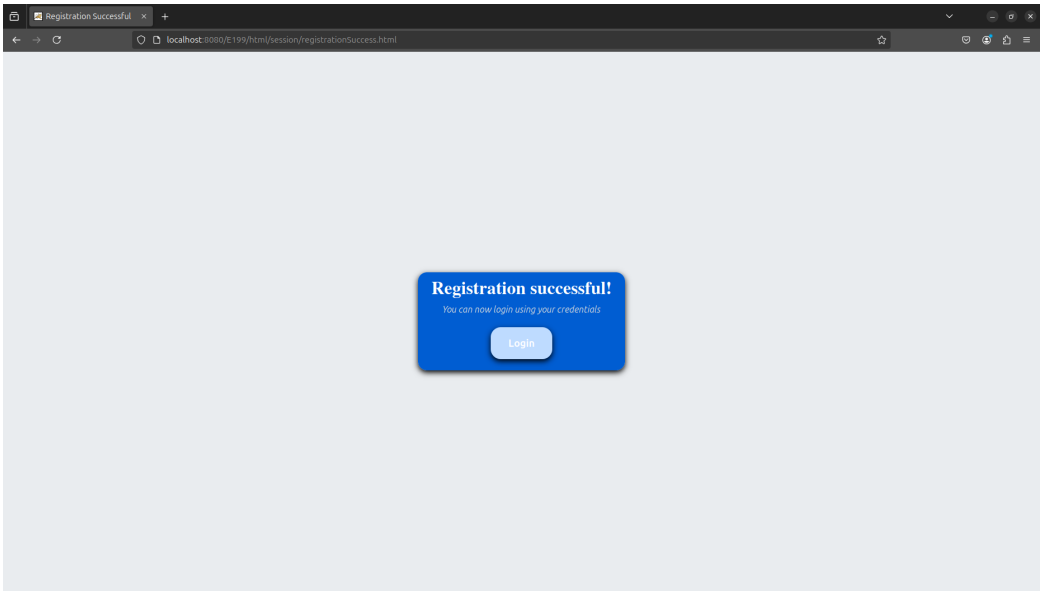

Figure 14: Successful user registration feedback

The user will be presented with a login button, which they can use to access the login screen and login with the appropriate user type and valid credentials

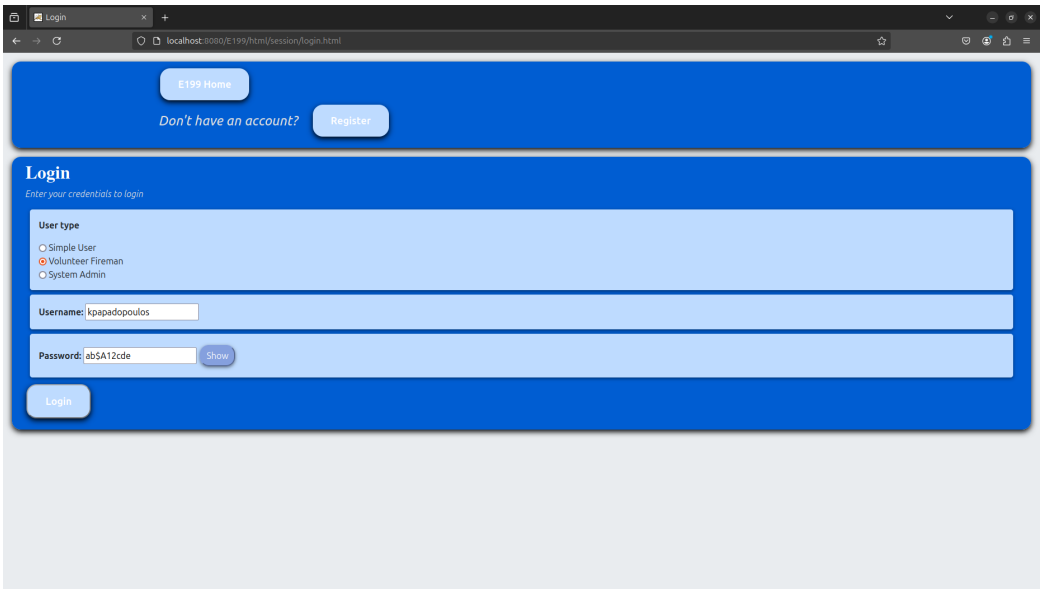

Figure 15: User logging in as a volunteer

Having logged in with valid credentials, the user will be directed to the dashboard. Below is a screenshot displaying the user information

Figure 16: Volunteer Dashboard displaying personal information of a volunteer

From the dashboard, the user can perform other operations such as viewing a list of incidents and chatting.